CLAIMS

I claim:

1. A method comprising:

pruning local graphs representing local problems, the local problems corresponding to separately compilable components in a software program, each of the local graphs having edges and vertices, each edge having a transfer function, each vertex having a value, values of each of the local graph forming a lattice under a partial ordering.

2. The method of claim 1 wherein pruning the local graphs comprising:

associating a use attribute to each one of the vertices in each of the local graphs, the use attribute being asserted for each vertex reachable from a named vertex;

associating an affect attribute to each one of the vertices in each of the local graphs, the affect attribute is asserted for a vertex if a named vertex is reachable from the former vertex; and

pre-solving a subgraph of each of the local graphs, the subgraph including subgraph edges, each of the subgraph edges connecting a tail vertex to a head vertex, the tail vertex having a negated use attribute.

3. The method of claim 2 wherein pruning the local graphs further comprising:

shrinking the local graphs.

4. The method of claim 3 further comprising solving a global problem to optimize a recompilation of the separately compilation components by an inter-procedural analysis (IPA) solver, the global problem being represented by a global graph formed from the pruned local graphs.

5. The method of claim 4 wherein pruning the local graphs further comprising:

determining final edges and vertex values of the local graphs to be sent to IPA solver; and

sending the final edges and vertex values to the IPA solver, the final edges and vertex values forming the pruned local graphs.

6. The method of claim 2 wherein associating the use attribute comprises:

negating use attributes for all vertices in the local graph; and

invoking a mark use operation on u for each named vertex u in the local graph.

7. The method of claim 6 wherein invoking the mark use operation on u comprises:

asserting the use attribute associated with u if the use attribute is negated; and

recursively invoking the mark use operation on v for each edge connecting the named vertex u to a vertex v.

8. The method of claim 2 wherein associating the affect attribute comprises:

negating use attributes for all vertices in the local graph;

invoking a mark affect operation on y for each named vertex y in the local graph.

9. The method of claim 8 wherein invoking the mark affect operation on y comprises:

asserting the use attribute associated with y if the use attribute is negated; and

recursively invoking the mark affect operation on x for each edge connecting the vertex x to a named vertex y.

10. The method of claim 2 wherein pre-solving the subgraph comprises:

finding a greatest fix-point solution to the subgraph.

11. The method of claim 3 wherein shrinking comprises:

removing an incoming edge having a head value of a lattice-bottom.

12. The method of claim 3 wherein shrinking further comprises:

transforming a subgraph having first and second edges, the first and second edges having first and second functions, the first edge connecting a first vertex to an anonymous vertex having a value v, the second edge connecting the anonymous vertex to a second vertex having a value w.

13. The method of claim 12 wherein transforming comprises:

removing the anonymous vertex;

removing first and second edges;

adding a third edge having a third function and connecting the first and second vertices, the third function being combined by the first and second functions; and changing value of the second vertex to a lattice meet of the second function of the value v and the value w.

14. The method of claim 5 wherein determining the final edges and vertex values comprises:

determining each of the final edges as edge having asserted use and affect attributes for tail and head vertices, respectively; and

eliding each of the vertex values having a top value.

15. A computer program product comprising:

a machine useable medium having computer program code embedded therein, the computer program product having:

computer readable program code to prune local graphs representing local problems, the local problems corresponding to separately compilable components in a software program, each of the local graphs having edges and vertices, each edge having a transfer function, each vertex having a value, values of each of the local graph forming a lattice under a partial ordering.

16. The computer program product of claim 15 wherein the computer readable program code to prune the local graphs comprising:

computer readable program code to associate a use attribute to each one of the vertices in each of the local graphs, the use attribute being asserted if there is an edge connecting a named vertex to the each one of the vertices;

computer readable program code to associate an affect attribute to each one of the vertices in each of the local graphs, the affect attribute is asserted if there is an edge connecting the each one of the vertices to a named vertex; and

computer readable program code to pre-solve a subgraph of each of the local graphs, the subgraph including subgraph edges, each of the subgraph edges connecting a tail vertex to a head vertex, the tail vertex having a negated use attribute.

17. The computer program product of claim 16 wherein the computer readable program code to prune the local graphs further comprising:

computer readable program code to shrink the local graphs.

18. The computer program product of claim 15 further comprising:

computer readable program code to solve a global problem to optimize a recompilation of the separately compilation components by an inter-procedural analysis (IPA) solver, the global problem being represented by a global graph formed from the pruned local graphs.

19. The computer program product of claim 18 wherein the computer readable program code to prune the local graphs further comprising:

computer readable program code to determine final edges and vertex values of the local graphs to be sent to IPA solver; and

computer readable program code to send the final edges and vertex values to the IPA solver, the final edges and vertex values forming the pruned local graphs.

20. The computer program product of claim 16 wherein the computer readable program code to associate the use attribute comprises:

computer readable program code to negate use attributes for all vertices in the local graph;

computer readable program code to invoke a mark use operation on u for each named vertex u in the local graph.

21. The computer program product of claim 19 wherein the computer readable program code to invoke the mark use operation on u comprises:

computer readable program code to assert the use attribute associated with u if the use attribute is negated; and

computer readable program code to recursively invoke the mark use operation on v for each edge connecting the named vertex u to a vertex v.

22. The computer program product of claim 16 wherein the computer readable program code to associate the affect attribute comprises:

computer readable program code to negate use attributes for all vertices in the local graph; and

computer readable program code to invoke a mark affect operation on y for each named vertex y in the local graph.

23. The computer program product of claim 22 wherein the computer readable program code to invoke the mark affect operation on y comprises:

computer readable program code to assert the use attribute associated with y if the use attribute is negated; and

computer readable program code to recursively invoke the mark affect operation on x for each edge connecting the vertex x to a named vertex y.

24. The computer program product of claim 16 wherein the computer readable program code to pre-solve the subgraph comprises:

computer readable program code to find a greatest fix-point solution to the subgraph.

25. The computer program product of claim 17 wherein the computer readable program code to shrink comprises:

computer readable program code to remove an incoming edge having a head value of a lattice-bottom.

26. The computer program product of claim 17 wherein the computer readable program code to shrink further comprises:

computer readable program code to transform a subgraph having first and second edges, the first and second edges having first and second functions, the first edge connecting a first vertex to an anonymous vertex having a value v, the second edge connecting the anonymous vertex to a second vertex having a value w.

27. The computer program product of claim 26 wherein the computer readable program code to transform comprises:

computer readable program code to remove the anonymous vertex;

computer readable program code to remove first and second edges;

computer readable program code to add a third edge having a third function and connecting the first and second vertices, the third function being combined by the first and second functions; and

computer readable program code to change value of the second vertex to a lattice meet of the second function of the value v and the value w.

28. The computer program product of claim 19 wherein the computer readable program code to determine the final edges and vertex values comprises:

computer readable program code to determine each of the final edges as edge having asserted use and affect attributes for tail and head vertices, respectively; and

computer readable program code to elide each of the vertex values having a top value.

29. A system comprising:

a processor; and

a memory coupled to the processor to store instruction code, the instruction code, when executed by the processor, causing the processor to:

prune local graphs representing local problems, the local problems corresponding to separately compilable components in a software program, each of the local graphs having edges and vertices, each edge having a transfer function, each vertex having a value, values of each of the local graph forming a lattice under a partial ordering.

30. The system of claim 29 wherein the instruction code causing the processor to prune the local graphs causes the processor to:

associate a use attribute to each one of the vertices in each of the local graphs, the use attribute being asserted if there is an edge connecting a named vertex to the each one of the vertices;

associate an affect attribute to each one of the vertices in each of the local graphs, the affect attribute is asserted if there is an edge connecting the each one of the vertices to a named vertex; and

pre-solve a subgraph of each of the local graphs, the subgraph including subgraph edges, each of the subgraph edges connecting a tail vertex to a head vertex, the tail vertex having a negated use attribute.

31. The system of claim 30 wherein the instruction code causing the processor to prune the local graphs further causes the processor to:

shrink the local graphs.

32. The system of claim 31 wherein the instruction code further causing the processor to:

solve a global problem to optimize a recompilation of the separately compilation components by an inter-procedural analysis (IPA) solver, the global problem being represented by a global graph formed from the pruned local graphs.

33. The system of claim 32 wherein the instruction code causing the processor to prune the local graphs further causes the processor to:

determine final edges and vertex values of the local graphs to be sent to IPA solver; and send the final edges and vertex values to the IPA solver, the final edges and vertex values forming the pruned local graphs.

34. The system of claim 30 wherein the instruction code causing the processor to pre-solve the subgraph causes the processor to:

find a greatest fix-point solution to the subgraph.

35. The system of claim 31 wherein the instruction code causing the processor to shrink causes the processor to:

remove an incoming edge having a head value of a lattice-bottom.

36. The system of claim 35 wherein the instruction code causing the processor to shrink further causes the processor to:

transform a subgraph having first and second edges, the first and second edges having first and second functions, the first edge connecting a first vertex to an anonymous vertex having a value v, the second edge connecting the anonymous vertex to a second vertex having a value w.

37. The system of claim 36 wherein the instruction code causing the processor to transform causing the processor to:

remove the anonymous vertex;

remove first and second edges;

add a third edge having a third function and connecting the first and second vertices, the third function being combined by the first and second functions; and

change value of the second vertex to a lattice meet of the second function of the value v and the value w.

38. The system of claim 33 wherein the instruction code causing the processor to determine the final edges and vertex values causes the processor to:

determine each of the final edges as edge having asserted use and affect attributes for tail and head vertices, respectively; and

elide each of the vertex values having a top value.